Tugas Rangkum Pemrograman Berbabasis Objek Pertemuan 1-6

Fiza Fahri Giwanggoro

50422588

2IA11

Rangkuman Materi Perteman 1-6

Pengenalan PBO

Pemrograman berorientasi objek (Object Oriented Programming atau disingkat OOP) adalah paradigma pemrograman yang berorientasikan kepada objek yang merupakan suatu metode dalam pembuatan program, dengan tujuan untuk menyelesaikan kompleksnya berbagai masalah program yang terus meningkat. Pemrograman berorientasi objek ditemukan pada Tahun 1960, dimana berawal dari suatu pembuatan program yang terstruktur (structured programming). Metode ini dikembangkan dari bahsa C dan Pascal. Pemrograman berorientasi objek bekerja dengan baik ketika dibarengi dengan Objek-Oriented Analysis And Design Process (OOAD).

Pemrograman Berbasis Objek

- Menggabungkan fungsi dan data dalam kelas – kelas atau objek – objek
- Memiliki ciri Encapsulation (pengemasan), Inheritance (penurunan sifat) dan Polymorphism (perbedaan bentuk dan perilaku)
- Struktur program ringkas, cukup dengan membuat Objek dan class lalu bekerja berdasarkan object dan class tersebut.
- Kode program sangat re-usable. object dan class dapat digunakan berkali-kali, sehingga dapat menghemat space memori.
- Efektif digunakan untuk menyelesaikan masalah besar, karena OOP terdiri dari class-class yang memisahkan setiap kode program menjadi kelompok - kelompok kecil, sesuai dengan fungsinya
- Sulit diawal (karena harus membuat class) namun selanjutnya akan terasa mudah dan cepat
- Eksekusi lebih cepat karena dieksekusi bersamaan, program hanya mengatur Objek, properties dan method-nya saja

Pemrograman Terstruktur

- Memecah program dalam fungsi dan data
- Memiliki ciri Sequence (berurutan), Selection (pemilihan) dan Repetition (perulangan)
- Struktur program rumit karena berupa urutan proses dan fungsi-fungsi
- Re-use kode program kurang
- Efektif digunakan untuk menyelesaikan masalah kecil dan tidak cocok untuk menyelesaikkan masalah yang rumit, karena nantinya akan kesulitan menemukan solusi permasalahan ketika terjadi error

- Mudah diawal, namun kompleks diproses selanjutnya
- Eksekusi lebih lambat karena setiap perintah dikerjakan berurutan

Class Pada PBO

Class adalah cetak biru atau blueprint dari object. Class digunakan hanya untuk membuat kerangka dasar.

Contoh Program:

```
// Deklarasi Class
public class Kendaraan{
    // Ini Adalah Class, semua konstruktor, variabel, method berada disini
}
```

OBJEK dalam Pemrograman

Objek adalah entitas yang memiliki atribut, karakter dan kadang kala disertai kondisi. Objek mempresentasikan sesuai kenyataan seperti siswa, mempresentasikan dalam bentuk konsep seperti merek dagang, juga bisa menyatakan visualilasi seperti bentuk huruf (font).

Pada dasarnya ada dua karakteristik yang utama pada sebuah objek , yaitu :

1. Setiap objek memiliki atribut sebagai status yang kemudian akan disebut sebagai **state**.
2. Setiap objek memiliki tingkah laku yang kemudian akan disebut sebagai **behaviour**.

Dalam pengembangan perangkat lunak berorientasi objek, objek dalam perangkat lunak akan menyimpan **state-nya dalam variable** dan menyimpan informasi **tingkah laku (behaviour) dalam method atau fungsi-fungsi/prosedur**

Objek pada PBO

Untuk membuat objek, bisa menggunakan kata kunci new yang digunakan untuk membuat objek baru, selanjutnya menentukan 3 langkah untuk membuat sebuah objek yaitu, mendeklarasikan variable, membuat objek baru (Instansiasi) dan pemanggilan konstruktor.

Contoh Program:

```
public class Kendaraan{

    // Konstruktor Dengan Parameter
    public Kendaraan(String nama){
        System.out.println("Nama Kendaraannya Adalah "+ nama);
    }

    public static void main(String[]args){

        // Perintah untuk membuat objek jenis
        Kendaraan jenis = new Kendaraan("Pesawat Terbang");

    }

}

// Output = Nama Kendaraannya Adalah Pesawat Terbang
```

Inheritance

Pewarisan merupakan sebuah bentuk "penggunaan kembali" (*reusability*); dimana **class** baru dibuat dari **class** yang pernah ada yang (biasanya) ditambah fasilitasnya. Setiap **class** turunan dapat menjadi **class** pokok (induk) untuk **class** turunan yang akan datang.

- Pewarisan tunggal (*single inheritance*) merupakan pewarisan dari satu **class** pokok (induk).
- Pewarisan ganda (*multiple inheritance*) merupakan pewarisan dari dua atau lebih **class** pokok.

Class yang mewariskan disebut dengan **superclass / parent class / base class**, sedangkan class yang mewarisi (class yang baru) disebut dengan **subclass / child class / derived class**. Untuk menerapkan inheritance, gunakan statement "**extends**". Keyword "**super**" digunakan oleh subclass untuk memanggil constructor, atribut dan method yang ada pada superclass-nya.

Contoh untuk memanggil constructor milik superclass-nya :

```
super()

super(parameter)
```

Contoh untuk memanggil atribut dan method milik superclass-nya :

```
super.namaAtribut
super.namaMethod(parameter)
```

Encapsulation

Enkapsulasi (encapsulation) merupakan cara untuk melindungi property (atribut) / method tertentu dari sebuah kelas agar tidak sembarangan diakses dan dimodifikasi oleh suatu bagian program.

Accessors

Cara untuk melindungi data yaitu dengan menggunakan access modifiers (hak akses). Ada 4 hak akses yang tersedia, yaitu default, public, protected, private

No	Modifier	Pada class dan interface	Pada method dan variabel
1	Default (tidak ada modifier)	Dapat diakses oleh yang sepaket	Diwarisi oleh subkelas dipaket yang sama, dapat diakses oleh method-method yang sepaket
2	Public	Dapat diakses dimanapun	Diwarisi oleh subkelasnya, dapat diakses dimanapun
3	Protected	Tidak bisa diterapkan	Diwarisi oleh subkelasnya, dapat diakses oleh method-method yang sepaket
4	private	Tidak bisa diterapkan	Tidak dapat dikases dimanapun kecuali oleh method-method yang ada dalam kelas itu sendiri

Aksesabilitas	public	private	protected	default
Dari kelas yang sama	Ya	Ya	Ya	Ya
Dari sembarang kelas dalam paket yang sama	Ya	Tidak	Ya	Ya
Dari sembarang kelas di luar paket	Ya	Tidak	Tidak	Tidak
Dari subkelas dalam paket yang sama	Ya	Tidak	Ya	Ya
Dari subkelas di luar paket	Ya	Tidak	Ya	Tidak

Overloading

Overloading adalah diperbolehkannya dalam sebuah class memiliki lebih dari satu nama function/method yang sama tetapi memiliki parameter/argument yang berbeda.

```java
public class Overloading {
    public void Tampil(){
        System.out.println("I love Java");
    }
    public void Tampil(int i){
        System.out.println("Method dengan 1 parameter = "+i);
    }
    public void Tampil(int i, int j){
        System.out.println("Method dengan 2 parameter = "+i+" & "+j);
    }
    public void Tampil(String str){
        System.out.println(str);
    }

    public static void main(String a[]){
        Overloading objek = new Overloading();
        objek.Tampil();
        objek.Tampil(8);
        objek.Tampil(6,7);
        objek.Tampil("Hello world");
    }
}
```

Overriding

Overriding method adalah kemampuan dari subclass untuk memodifikasi method dari superclass-nya, yaitu dengan cara menumpuk (mendefinisikan kembali) method superclass-nya. Contoh overriding method dapat dilihat pada subclass “Mobil” yang mendefinisikan kembali method keterangan() dan hapus() dari class “Kendaraan”.